Chapter

7

Optimization of Staged and Discrete Processes

7.1 Dynamic Programming

Multi-stage decision process arises during the study of different mathematical problems; the dynamic programming technique was created for taking care of these sorts of problems. Consider a physical system, at any time *t*, the state of this system is found out by a set of quantities that is called state variables or state parameters. At certain times, we are call upon to make a decision that will have an effect on the state of the system. This time may be prescribed in advanced or the process itself determines the time. The transformations of the state variables are equivalent to these decisions, the selection of a decision being identical with the choice of a transformation. The future state is guided by the outcome of the preceding decision, which maximizes some function of the parameters that describe the final state.

There are many practical problems where sequential decisions are made at various points in time, at various points in space, and at various levels, say, for a system, for a subsystem, or even for a component level. The problems for which the decisions are required to make sequentially are termed as sequential decision problems. They are also called multistage decision problems because for these problems decisions are to be made at a number of stages. Dynamic programming is a mathematical technique, which is most suitable for the optimization of multistage decision problems. Richard Bellman in the early 1950s [Bellman, (1953)] developed this technique. During application of dynamic programming method, a multi-dimensional decision problem was decomposed into a series of single stage decision problems. In this way, an N-variable problem can be expressed as a series of N single-variable problems. These N number of single-variable problems can be solved sequentially. Most of the time, these single-variable subproblems are quite easier to solve compared to the original problem. This decomposition to N single-variable subproblems is accomplished in such a way that the optimal solution of the original N-dimensional problem can be attained from the optimal solutions of the N one-dimensional problems. It is essential to make a note that the methods used to solve these one-dimensional stages do not affect the overall result.

Any method can be used for this purpose; these methods may vary from an easy enumeration method to a relatively complicated technique like differential calculus or a nonlinear programming.

Classical optimization techniques can also be employed directly to solve the multistage decision problems. However there are some limitations, this is possible when small number of variables is there, the concerned functions should be continuous and continuously differentiable, and also the optimum points should not be positioned on the boundary line. Furthermore, the form of the problem needs to be quite simple such that the set of resultant equations can be solved by either analytical method or numerical method. To work out with somewhat more complex multistage decision problems we can use the nonlinear programming techniques. However, prior knowledge on the region of the global maxima or minima is required for their application, and the variables should also be continuous. For all these instances, the presence of stochastic variability changes the problem to a very complicated one and yields the problem unsolvable except by employing some sort of an approximation e.g., chance constrained programming. In contrast, the dynamic programming is able to solve the problems with discrete variables, and non-continuous, non-convex, and non-differentiable in nature. Usually it also considers the stochastic variability by a mere change of the deterministic procedure. There is a major drawback of the dynamic programming method that is recognized as the *curse of dimensionality*. Regardless of this weakness, it is quite appropriate for solving various types of complicated problems in diverse areas of decision-making.

Principle of optimality Irrespective of the initial state and initial decisions, the remaining decisions should formulate an optimal policy with regard to the state resulting from the first decisions [Bellman, (1954)].

7.1.1 Components of dynamic programming

A dynamic programming functional equation can be represented by the following equation

$$f(S) = \mathrm{opt}_{d \in D(S)} \left\{ R(S,d) \circ f\left(T(S,d)\right) \right\} \tag{7.1}$$

where S represents a state within a state space **S**, d signifies a decision selected from a decision space $D(S)$, $R(S, d)$ is called reward function (or decision cost), $T(S, d)$ is the transition or transformation function for the next-state, and "°" stands for a binary operator. In this chapter, only the discrete dynamic programming is discussed, where both the state space and decision space are discrete sets.

State The state S usually includes information on the series of decisions made until now. The state may be a complete sequence for some situations but, only partial information is satisfactory for other situations; for instance, when the set of all states can be divided into similar modules, all of them expressed by the last decision. For some uncomplicated problems, the extent of the sequence that is also called the stage at which the next decision is to be made, becomes sufficient. The state at the beginning that reflects the condition in which no decision has been made yet, is called the goal state and it is represented by S^*.

Decision Space The decision space $D(S)$ is defined by the set of possible or "eligible" choices for the next decision d. This decision space is a function of S (the state) within which the decision d is to be made. Constraints on probable transformations to the next-state from a state S can be implemented by properly restricting $D(S)$. The S is called a terminal state when there are no eligible decisions in state S i.e., $D(S) = 0$.

Objective Function The objective function f is a function of S. During the application of dynamic programming, we need to optimize this objective function. This is the optimal profit or cost resulting from creating a progression of decisions when within the state S, i.e., after creating the series of decisions correlated with S. The aim of a dynamic programming problem is to find $f(S)$ for the goal state S^*.

Reward Function The reward function R can be represented as a function of the state S and the decision d. Reward function is the cost or profit, which can be attributed to the subsequent decision d made in state S. This is also known as "Return Function". The reward $R(S, d)$ have to be separable from the costs or profits that are attributed to all other decisions. The value of $f(S^*)$, the objective function for the goal state, is the combination of the rewards for the entire optimal series of decisions starting from the goal state.

Transformation Function(s) The transition or transformation function (T) can be represented as a function of the state S and the decision d. This function identifies the next-state that results from making a decision d in state S. There may be more than one transformation function for a non-serial dynamic programming problem.

Operator The operator (°) is a binary operation, which enables us to connect the returns of various separate decisions. Usually this operator is either addition or multiplication. The operation should be associative whenever the returns of decisions are to be independent of the order in which they are made.

7.1.2 Theory of dynamic programming

As discussed above, the fundamental concept for the theory of dynamic programming is that of representing an optimal strategy as one finding out the required decision at every time in terms of the system's present state. The mathematical equation of dynamic programming problem has been developed based on this idea.

Mathematical formulation

In this section, we will formulate the mathematical equation of a deterministic discrete process as described by Bellman [Bellman, (1954)]. To demonstrate this kind of functional equation that appears from an application of the principle of optimality, let us start with a very simple example of a deterministic process in which the system is expressed at any time by a vector in M-dimensional space $p = (p_1, p_2, \ldots, p_M)$, constrained to be positioned within some region D. Let $T = \{T_k\}$, where k runs over a set that may be continuous, finite, or enumerable, be a set of transformations with the characteristic that $p \in D$ indicates that $T_k(p) \in D$ for all k.

Let us consider an N-stage process to be performed to maximize some scalar function, $R(p)$ of the final state. This function is called the N-stage return. The strategy is composed of a selection of N transformations, $P = (T_1, T_2, \ldots, T_N)$, yielding successively the states

$$p_1 = T_1(p)$$

$$p_2 = T_2(p_1) \tag{7.2}$$

....

$$p_N = T_N\left(p_{N-1}\right)$$

When, the region D is finite, if every $T_k(p)$ is continuous in p, and if $R(p)$ is a continuous function of p for $p \in D$, then there must exists an optimal policy. The maximum value of the return $R(p_N)$, which is decided by an optimal policy, only be a function of the number of stages N and the initial vector p. Then we can determine

$$f_N\left(p\right) = \underset{p}{\text{Max}}\, R\left(p_N\right) \tag{7.3}$$

which is the return achieved after N-stage by utilizing an optimal policy that starts from the initial state p.

To develop a functional equation for $f_N(p)$, we use the principle mentioned above. Suppose that we have chosen some transformation T_k, which is the result of our first decision, in this manner we are obtaining a new state $T_k(p)$. Therefore, by definition, the maximum return from the following $(N-1)$ stages is $f_{N-1}(T_k(p))$. It follows that k must now be chosen so as to maximize this. The result is the basic functional equation

$$f_N\left(p\right) = \underset{k}{\text{Max}}\, f_{N-1}\left(T_k\left(p\right)\right) \quad N = 1, 2, \ldots \tag{7.4}$$

It is obvious that a knowledge of any particular optimal policy, necessarily not unique, will produce $f_N(p)$ that is unique. On the other hand, given the series $\{f_N(p)\}$, all optimal policies may be developed.

7.1.3 Description of a multistage decision process

A single-stage decision process is represented by the rectangular block in Fig 7.1. This single-stage is a part of some multistage problem. The whole decision process is characterized by the input data (or input parameters), S, decision variables (X), and the output parameters (T) that represent the outcome achieved as an effect of making the decision. The input parameters and the output parameters are defined as input state variables and output state variables respectively. Lastly, we get an objective function or return (R) that evaluates the effectiveness of the decisions made and the output as the consequence of these decisions.

The output of single-stage decision process as represented in Fig. 7.1 is associated with the input parameters through a stage transformation function represented by

$$T = t\left(X, S\right) \tag{7.5}$$

As the system's input state variable influences the decisions made, then we can express the return function as

$$R = r\left(X, S\right) \tag{7.6}$$

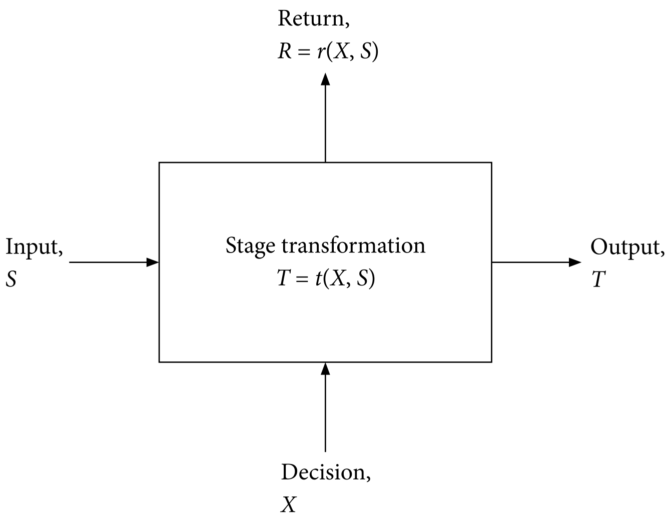

Fig. 7.1 Single-stage decision process

Schematic representation of a sequential multistage decision process is given in Fig. 7.2. Due to some convenience, the stages are labeled in decreasing order like n, $n - 1$, …, i, …, 2, 1. Therefore, the input state vector for the ith stage is indicated by S_{i+1} accordingly the output state vector as S_i. For a serial system, it is quite obvious that the output from $i + 1$ stage should be identical with the input to the next stage i. Thus, the state transformation and return functions for the ith stage can be expressed as

$$S_i = t_i\left(S_{i+1}, X_i\right) \tag{7.7}$$

$$R_i = r_i\left(S_{i+1}, X_i\right) \tag{7.8}$$

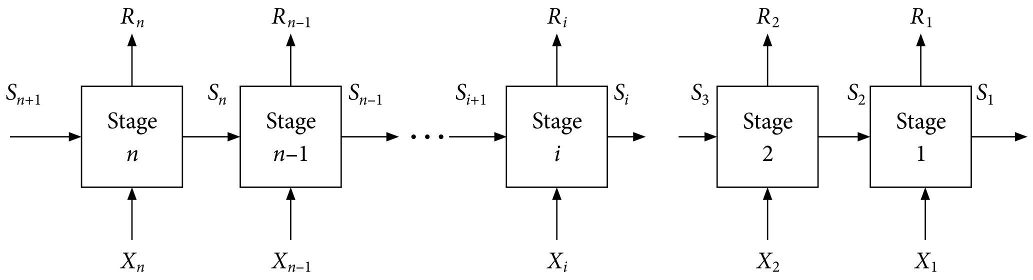

Fig. 7.2 Multi-stage decision process

where X_i represents the vector of decision variables for ith stage. The state transformation equations represented by Eq. (7.7) are also identified as design equations.

The purpose of a multistage decision problem is to determine a series $X_1, X_2, \ldots, X_n$ in order to optimize $f(R_1, R_2, \ldots, R_n)$ which is some function of the individual stage returns, and satisfy

Eqs (7.7) and (7.8). If a given multistage problem is solvable by using the dynamic programming are determined by the characteristic of the n-stage return function (f).

The monotonicity and separability of the objective function are required as this method runs as a decomposition technique. The objective function will be separable when it is possible to express the objective function as the composition of the individual stage returns. The additive objective functions satisfy this requirement:

$$f = \sum_{i=1}^{n} R_i = \sum_{i=1}^{n} R_i\left(X_i, S_{i+1}\right) \tag{7.9}$$

where X_i are real, and when the objective functions are multiplicative

$$f = \prod_{i=1}^{n} R_i = \prod_{i=1}^{n} R_i\left(X_i, S_{i+1}\right) \tag{7.10}$$

where X_i are nonnegative and real. Conversely, the following objective function given by Eq. (7.11) is not separable:

$$f = \left[R_1\left(X_1,S_2\right)+R_2\left(X_2,S_3\right)\right]\left[R_2\left(X_2,S_3\right)+R_3\left(X_3,S_4\right)\right]\left[R_3\left(X_3,S_4\right)+R_4\left(X_4,S_5\right)\right] \tag{7.11}$$

There are various practical problems in the field of chemical engineering that fulfill the condition of separability. For a monotonic objective function, values of all a and b that show

$$R_i\left(X_i = \mathbf{a}, S_{i+1}\right) \geq R_i\left(X_i = b, S_{i+1}\right) \tag{7.12}$$

the given inequality in Eq. (7.13) is satisfied:

$$f\left(X_n, X_{n-1}, \ldots, X_{i+1}, X_i = \mathbf{a}, X_{i-1}, \ldots, X_1, S_{n+1}\right) \geq f\left(X_n, X_{n-1}, \ldots, X_{i+1}, X_i = b, X_{i-1}, \ldots, X_1, S_{n+1}\right)$$
$$i = 1, 2, \ldots, n \tag{7.13}$$

The application of DP method in chemical engineering can be illustrated with the following example.

Example 7.1

Optimal Replacement of equipment

The dynamic programming method can be discussed with this example of equipment replacement in a chemical industry. In this example, time (each year) is considered as a discrete stage.

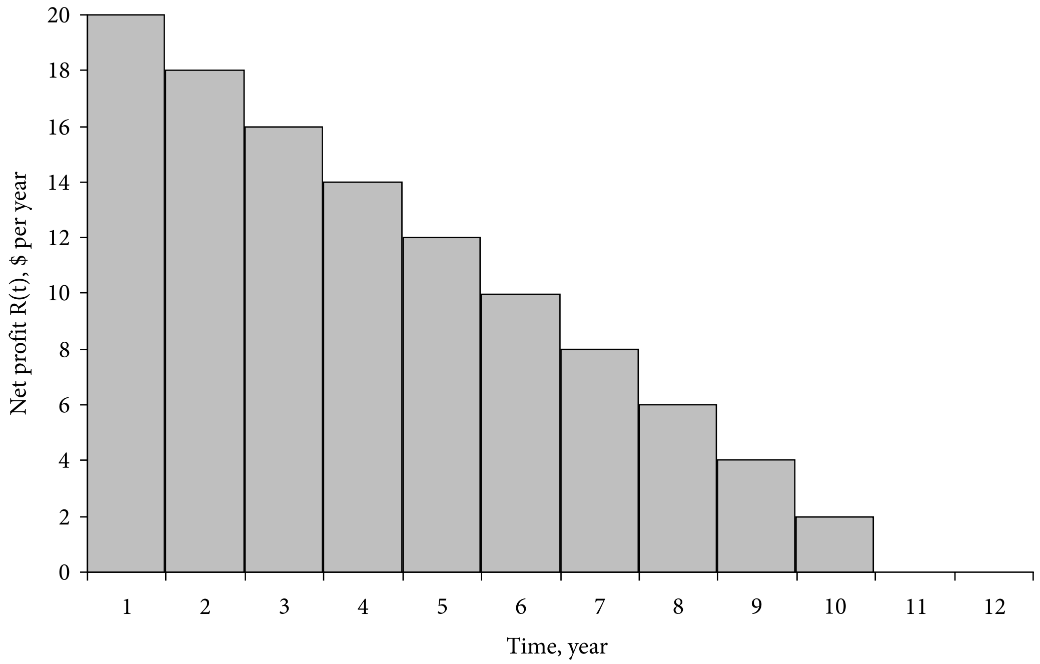

Fig. 7.3 The annual net profit vs. time plot

Figure 7.3 shows the data of the annual net profit for the operation of a continuous distillation unit over a time span of 12 year, where 10th year shows the break-even point for this distillation process. The replacement cost of this distillation unit with a new one employing up to date technology is considered to be the same as the net profit made during the 1st year of operation of the new unit, i.e., $20,000. An annual assessment is conducted at the starting of every year, and a decision is taken whether to continue with the old unit or replace it with a new contemporary model to get the maximum profit over a period of five years. At this moment, the distillation unit is four years old, and it is required to find out the best alternative now and for each of the following four years to maximize the profit, i.e., establish the optimal replacement strategy. This procedure starts with the consideration of the possible decisions to be made at the beginning of the fifth year (end of the fourth year), i.e., either continue with the process or replace it. Stage 1 is the time period from the beginning to the end of the fifth year. The algorithm for dynamic programming at stage one can be represented as:

$$f_1(S_1) = \underset{d_1}{\text{Max}}\left[R_1(S_1, d_1)\right] = \underset{\substack{K\\ or\\ R}}{\text{Max}}\begin{cases} R_1(t) \\ -20 + R(0) = 0 \end{cases} \tag{7.14}$$

where d_1 represents the decision, keep (K) or replace (R) the distillation unit to maximize the profit, $R_1(S_1, d_1)$. Moreover the profit from the unit depends on the age of this process, the state variable S_1. Figure 7.4 shows the stages of the dynamic programming. The optimum decisions are shown in Table 7.1 for stage one as a function of the state variable, and they are to keep the process operating. At the beginning of the fifth year, the range of state variables goes from a process one year old to having a process ten years old. Similarly, for a ten year old process there is a tie between keeping the old distillation unit and replacing it with a new one, and the decision made here is the one that is easier, i.e., keep the process operating. The values shown in Table 7.1 were obtained from Fig. 7.3. The output state variable from stage one S_0 is the age of the process at the end of the year.

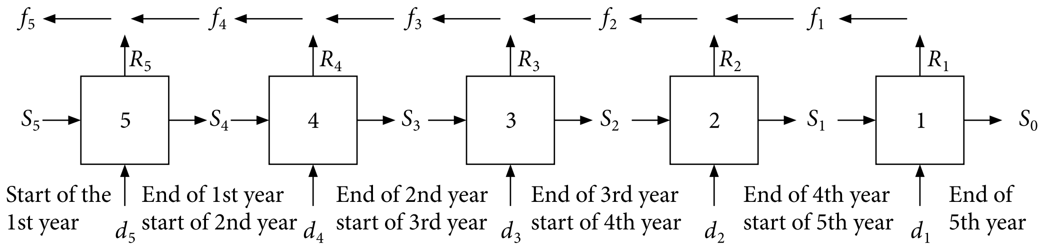

Fig. 7.4 Representation of 5 stage dynamic programming

Table 7.1 Values of the decision variables (as per Eq. (7.14))

S_0	2	3	4	5	6	7	8	9	10	11
d_1	K	K	K	K	K	K	K	K	K	K
f_1	18	16	14	12	10	8	6	4	2	0
S_1	1	2	3	4	5	6	7	8	9	10

At stage 2, we have taken the optimal decisions, which maximize the sum of the return at stage two and the optimal return from stage 1 for a range of the state variable values at stage 2. The algorithm of dynamic programming at stage 2 is:

$$f_2(S_2) = \underset{d_2}{\text{Max}}\left[R_2(S_2, d_2) + f_1(S_1)\right] = f_2(t) = \underset{\substack{K\\or\\R}}{\text{Max}}\begin{cases} R_2(t) + f_1(t+1) \\ -20 + R(0) + f_1(1) = 18 \end{cases} \tag{7.15}$$

If the decision is to keep (K) and continue with the same distillation unit, $f_2(t)$, the optimal return, is the sum of the return during the 4th year for a process t years old, $R_2(t)$, and $f_1(t+1)$, the optimal return from stage 1 for a process whose age is in $t+1$. If the decision is to replace (R) the distillation unit, the optimal return $f_2(t)$ is the sum of the cost of a new distillation unit, –20, the profit from operating a new unit for a year, $R(0) = 20$ and $f_1(1)$, the optimal return from stage 1 for a process one year old. The optimal decisions are given in Table 7.2 at stage two for a process whose age can be from 1 to 7 years, S_2. It is obvious from the figure that the optimal decisions are to continue to the operation of old distillation unit if its age is from 1 to 5 years old. However, if the unit is 6 years old or older, the profit will be higher for the 4th and 5th years if the old unit is replaced with a new one.

Table 7.2 Values of the decision variables (as per Eq. (7.15))

S_1	2	3	4	5	6	1	1
d_2	K	K	K	K	K	R	R
f_2	34	30	26	22	18	18	18
S_2	1	2	3	4	5	6	7

The similar method is employed at stage three to determine the maximum profit for the 3rd, 4th, and 5th years. The algorithm for dynamic programming is represented as

$$f_3(S_3) = f_3(t) = \underset{\substack{K\\or\\R}}{\text{Max}} \begin{cases} R_3(t) + f_2(t+1) \\ -20 + R(0) + f_2(1) = 34 \end{cases} \tag{7.16}$$

The optimal decisions for stage three are given in Table 7.3. At this stage, the optimal decisions are to continue the operation of the distillation unit if its age is from 1 to 3 years old; and if it is older, the maximum profit over the 3 year period will be achieved by replacing the old unit with a new one.

Table 7.3 Values of the decision variables (as per Eq. (7.16))

S_2	2	3	4	1	1
d_3	K	K	K	R	R
f_3	48	42	36	34	34
S_3	1	2	3	4	5

Continuing to stage 4, we need to repeat the procedure to determine the maximum profit for the 4 year period. The algorithm of the dynamic programming to determine the optimal decisions for various age processes (the state variable) is represented by the following equation:

$$f_4(t) = \underset{\substack{K\\or\\R}}{\text{Max}} \begin{cases} R_4(t) + f_3(t+1) \\ -20 + R(0) + f_3(1) = 48 \end{cases} \tag{7.17}$$

As shown in Table 7.4, the optimal decisions are to keep and run the distillation unit if it is from 1 to 3 years older and if it is older, replace the unit with a new one.

Table 7.4 Values of the decision variables (as per Eq. (7.17))

S_3	2	3	4	1	1
d_4	K	K	K	R	R
f_4	60	52	48	48	48
S_4	1	2	3	4	5

The results for the 5th or final stage are obtained by using the similar method as before. However, it is required to consider only one value of the state variable for the existing 4 years old distillation unit. The algorithm of the dynamic programming is:

$$f_5(t) = \underset{\substack{K \\ or \\ R}}{\text{Max}} \begin{cases} R_5(t) + f_4(t+1) \\ -20 + R(0) + f_4(1) = 60 \end{cases} \tag{7.18}$$

Table 7.5 Values of the decision variables (as per Eq. (7.18))

S_3	2	3	4	5	1
d_4	K	K	K	K	R
f_4	70	64	62	60	60
S_4	1	2	3	4	5

As described by Eq. (7.18) and Table 7.1–7.5 the maximum profit for the period of five years is \$60,000 when the process is 4 year old. The optimal decisions are taken as continue the process for the 1st year and at the beginning of the second year replace it with a new one and to operate this new process for the remaining three years. Other cases are also described above example that were found by using the dynamic programming algorithm for processes that are 1, 2, 3 and 5 years old. For example, the maximum profit would be \$70,000 for a 1-year-old process. The optimal decisions for this process would be to continue with the same process for the five year period. Whenever the process becomes 5 years old, the maximum profit would be \$60,000; and the optimal decisions would be to replace the existing process with a new one and run this new process for the 5 year period. As a result, the dynamic programming algorithm produced other probably useful information without considerable additional computational load.

7.2 Integer and Mixed Integer Programming

As MILP has already been discussed in chapter 6 (6.2.4), we will focus on the mixed integer nonlinear programming in this section. In chemical engineering, there are many applications of nonlinear optimization problems that involve integer or discrete variables besides the continuous variables. These types of optimization problems are denoted as Mixed-Integer Nonlinear Programming (MINLP) problems. The integer variables can be utilized for modeling, for example, series of events, alternative candidates, existence or nonexistence of units (in their zero-one representation), number of trays for distillation column design etc. Discrete variables can also be used for model development, for example, different equipment sizes like standard pipe diameter, standard capacity of pump etc.

The coupling of the continuous domain and integer variables along with their associated nonlinearities make the category of MINLP problems quite challenging from the theoretical, algorithmic, and computational standpoint. Despite this challenging nature, there exists a broad spectrum of applications that can be formulated as mixed-integer nonlinear programming problems. MINLPs have been adopted in different fields, including the financial sector and the process industry, management science, engineering, and operations research sectors. Process engineers are using this method for process synthesis which include: (i) the synthesis of heat recovery networks (Floudas and Ciric, 1989); (ii) membrane system for multicomponent gas mixtures (Qi and Henson, 2000) (iii) the synthesis of complex reactor networks (Kokossis and Floudas, 1990); (iv) the synthesis of utility systems (Kalitventzeff and Marechal, 1988); and the synthesis of total process systems (Kocis and Grossmann, 1988). An excellent review of the mixed integer nonlinear optimization frameworks and applications in Process Synthesis are provided in Grossmann (1990). Algorithmic advances for logic and global optimization in Process Synthesis are reviewed in Floudas and Grossmann (1994). Key applications of MINLP approaches have also emerged in the area of Design, Scheduling, and Planning of Batch Processes in chemical engineering and include: (i) the design of multiproduct plants (Grossmann and Sargent, 1979; Birewar and Grossmann, 1990). Excellent recent reviews of the advances in the design, scheduling, and planning of batch plants can be found in Reklaitis (1991), and Grossmann *et al.* (1992).

7.2.1 Formulation of MINLP

The primary objective of this section is to present the general formulation of MINLP problems, discuss the difficulties, and give an overview of the algorithmic approaches developed for these problems.

Mathematical description

The general form of a MINLP is

$$\min_{x,y} f(x,y) \tag{7.19a}$$

$$\text{subject to } h(x,y)=0 \tag{7.19b}$$

$$g(x,y)\le 0 \tag{7.19c}$$

$$x \in X \subseteq \Re^n \tag{7.19d}$$

$$y \in Y \quad \text{integer} \tag{7.19e}$$

Here, $f(x, y)$ is a nonlinear objective function (e.g., annualized total cost, profit, energy consumption), x represents a vector of n continuous variables (e.g., flow, pressure, composition, temperature), and y is a vector of integer variables (e.g., alternative solvents or materials, number of units in HEN, and reactor network, standard pipe diameter); $h(x, y) = 0$ denote the m equality constraints (e.g., mass, and energy balances, equilibrium relationships); $g(x, y) \le 0$ are the p inequality constraints (e.g., specifications on purity of distillation products, environmental regulations such as maximum limit of pollutant discharge, feasibility constraints in heat recovery systems, logical constraints).

Remark 1

The integer variables *y* with given lower and upper bounds,

$$y^L \leq y \leq y^U \tag{7.20}$$

This variable *y* can be expressed through 0–1 variables (i.e., binary) denoted as *z*, by the following formula:

$$y = y^L + z_1 + 2z_2 + 4z_3 + \ldots + 2^{N-1} z_N \tag{7.21}$$

where *N* is the minimum number of 0–1 variables needed. This minimum number is given by

$$N = 1 + INT\left\{\frac{\log\left(y^U - y^L\right)}{\log 2}\right\} \tag{7.22}$$

where *INT* function truncates its real argument to an integer value. This approach however may not be practical when the bounds are large.

Then, formulation (7.19a–7.19e) can be written in terms of 0–1 variables:

$$\min_{x,y} f\left(x, y\right) \tag{7.23a}$$

$$\text{subject to } h\left(x, y\right) = 0 \tag{7.23b}$$

$$g\left(x, y\right) \leq 0 \tag{7.23c}$$

$$x \in X \subseteq \Re^n \tag{7.23d}$$

$$y \in Y = \{0,1\}^q \tag{7.23e}$$

where *y* now is a vector of *q*, 0 – 1 variables (e.g., existence of a process unit ($y_i = 1$) or non-existence ($y_i = 0$)). We will focus on (7.23a–7.23e) in the majority of the subsequent developments.

Solution Approaches

A collection of MINLP algorithms has given below. These algorithms are developed for solving MINLP models of the form (7.23a–7.23e) or restricted classes of (7.23a–7.23e) includes the following in chronological order of development:

1. Generalized Benders Decomposition, GBD (Geoffrion, 1972);
2. Branch and Bound, BB (Beale, 1977);
3. Outer Approximation, OA (Duran and Grossmann, 1986);

4. Outer Approximation with Equality Relaxation, OA/ER (Kocis and Grossmann, 1987);
5. Outer Approximation with Equality Relaxation (Viswanathan and Grossmann, 1990)
6. Generalized Outer Approximation, GOA (Fletcher and Leyffer, 1994)
7. Generalized Cross Decomposition, GCD (Holmberg, 1990);

The following section elucidates the Generalized Benders Decomposition method with problem formulation.

7.2.2 Generalized Benders Decomposition

The decomposition method was first developed by Benders (Benders, 1962) to solve the mixed-variable programming problems. After that, the Generalized Benders Decomposition (GBD) was formulated for MINLP. However, some limitations were identified regarding the convexity and other properties of the functions involved. In a pioneering work of Geoffrion (1972) on the Generalized Benders Decomposition (GBD) two sequences of updated upper (nonincreasing) and lower (non-decreasing) bounds are created that converge within ε, in a finite number of iterations. The upper bounds correspond to solving subproblems in the x variables by fixing the y variables, whereas the lower bounds are based on duality theory. A review paper by Floudas *et al.* (Floudas *et al.* 1989) enlightened the potential applications of this technique for chemical process design. It also suggests a computational implementation for identifying global optimum point in nonconvex NLP and MINLP problems.

GBD requires the successive solution of an associated MIP problem. The algorithm decomposes the MINLP into a NLP subproblem with the discrete variables fixed and a linear MIP master problem. The major difference between OA and GBD is in the definition of the MIP master problem. OA relies on linearizations or tangential planes that reduce each subproblem effectively to a smaller feasible set, while the master MIP problem for a GBD is represented by a dual representation of the continuous space.

7.2.2.1 Formulation

Geoffrion (1972) generalized the approach proposed by Benders (1962), for exploiting the structure of mathematical programming problems (7.23a–7.23e), to the class of optimization problems stated as

$$\min_{x,y} f(x,y) \tag{7.23a}$$

$$\text{subject to } h(x,y)=0 \tag{7.23b}$$

$$g(x,y)\le 0 \tag{7.23c}$$

$$x\in X\subseteq \Re^n \tag{7.23d}$$

$$y\in Y=\{0,1\}^q \tag{7.23e}$$

under the following conditions:

C1 X is a nonempty, convex set and the functions

$$f : \Re^n \times \Re^q \to \Re \tag{7.24a}$$

$$g : \Re^n \times \Re^q \to \Re^p \tag{7.24b}$$

are convex for each fixed $y \in Y = \{0,1\}^q$, while the function $h : \Re^n \times \Re^l \to \Re^m$ are linear for each fixed $y \in Y = \{0,1\}^q$.

C2 The set

$$Z_y = \left\{ z \in \Re^p : h(x,y) = 0, g(x,y) \le z \text{ for some } x \in X \right\} \tag{7.25}$$

is closed for each fixed $y \in Y$

C3 For every fixed $y \in Y \cap V$, where

$$V = \left\{ \mathrm{y} : h(x,y) = 0, g(x,y) \le 0 \text{ for some } x \in X \right\} \tag{7.26}$$

one of the following two conditions holds:

i. the resulting problem (7.23a–7.23e) possesses a finite solution and has an optimal multiplier vector for the equalities and inequalities.

ii. the resulting problem (7.23a–7.23e) is unbounded, that is, its objective function value goes to $-\infty$.

The basic idea in Generalized Benders Decomposition (GBD) is the generation of an upper bound and a lower bound at each iteration for the required solution of the MINLP model. The primal problem gives us the upper bound, whereas the lower bound results from the master problem. The primal problem is similar to the problem (7.23–7.23e) with fixed y-variables (i.e., it is in the x space only), and the solution gives information about the upper bound and the Lagrange multipliers associated with the equality and inequality constraints. The master problem is obtained by the use of nonlinear duality theory, makes use of the Lagrange multipliers obtained in the primal problem. The solution of this master problem gives information about the lower bound, and the next set of fixed y-variables to be used subsequently in the primal problem. As the iterations continue, it is noticed that the sequence of updated upper bounds is non-increasing, the sequence of lower bounds is non-decreasing, and that the sequences converge in a finite number of iterations.

7.2.2.2 Theoretical development

This section presents the theoretical development of the Generalized Benders Decomposition (GBD). The primal problem is analyzed first for the feasible and infeasible cases. Subsequently, the theoretical analysis for the derivation of the master problem is presented.

The primal problem

The primal problem is obtained by fixing the y variables to a particular 0–1 combination, which can be denoted as y^k where k stands for the iteration counter. The formulation of the primal problem $P(y^k)$, at iteration k is

$$\left.\begin{aligned} &\min_x f\left(x, y^k\right) \\ \text{s.t.}\quad &h\left(x, y^k\right) = 0 \\ &g\left(x, y^k\right) \le 0 \\ &x \in X \subseteq \Re^n \end{aligned}\right] \qquad P\left(y^k\right) \tag{7.27}$$

Remark 1 Note that due to conditions C1 and C3(i), the solution of the primal problem $P(y^k)$ is its global solution.

We will distinguish the two cases of (i) feasible primal, and (ii) infeasible primal, and describe the method of analysis for each case separately.

Case (i): Feasible Primal

When the primal problem at iteration k is feasible, then its solution gives information on x_k, $f(x^k, y^k)$, which is the upper bound, and the optimal multiplier vectors λ^k, μ^k for the equality and inequality constraints. Subsequently, using this information we can formulate the Lagrange function as

$$L\left(x, y, \lambda^k, \mu^k\right) = f\left(x, y\right) + \lambda^{k^T} h\left(x, y\right) + \mu^{k^T} g\left(x, y\right) \tag{7.28}$$

Case (ii): Infeasible Primal

If the primal is detected by the NLP solver to be infeasible, then we consider its constraints

$$h\left(x, y^k\right) = 0 \tag{7.29a}$$

$$g\left(x, y^k\right) \le 0 \tag{7.29b}$$

$$x \in X \subseteq \Re^n \tag{7.29c}$$

where the set X, for instance, consists of lower and upper bounds on the x variables. To identify a feasible point we can minimize an l_1 or l_∞ sum of constraint violations. An l_1-minimization problem can be formulated as

$$\min_{x \in X} \sum_{i=1}^{p} \alpha_i \tag{7.30a}$$

$$\text{subject to}\quad h\left(x, y^k\right) = 0 \tag{7.30b}$$

$$g_i\left(x, y^k\right) \le \alpha_i, \quad i = 1, 2, \ldots, p \tag{7.30c}$$

$$\alpha_i \ge 0, \quad i = 1, 2, \ldots, p \tag{7.30d}$$

Note that if $\sum_{i=1}^{p} \alpha_i = 0$, then a feasible point has been determined.

Also note that by defining as

$$\alpha^+ = \max\left(0, \alpha\right) \tag{7.31}$$

and

$$g_i^+\left(x, y^k\right) = \max\left[0, \mathrm{g}_i\left(x, y^k\right)\right] \tag{7.32}$$

the l_1-minimization problem is stated as

$$\min_{x \in X} \sum_{i=1}^{P} g_i^+ \tag{7.33a}$$

$$\text{subject to } h\left(x, y^k\right) = 0 \tag{7.33b}$$

An l_∞-minimization problem can be stated similarly as:

$$\min_{x \in X} \max_{1,2,\ldots P} \mathrm{g}_i^+\left(x, y^k\right) \tag{7.34a}$$

$$\text{subject to } h\left(x, y^k\right) = 0 \tag{7.34b}$$

Alternative feasibility minimization approaches aim at keeping feasibility in any constraint residual once it has been established. An l_1-minimization in these approaches takes the form:

$$\min_{x \in X} \sum_{i \in I'} g_i^+\left(x, y^k\right) \tag{7.35a}$$

$$\text{subject to } h\left(x, y^k\right) = 0 \tag{7.35b}$$

$$g_i\left(x, y^k\right) \le 0, \quad i \in I \tag{7.35c}$$

where I is the set of feasible constraints; and I' is the set of infeasible constraints. Other methods seek feasibility of the constraints one at a time whilst maintaining feasibility for inequalities indexed by $i \in I$. This feasibility problem is formulated as

$$\min_{x \in X} \sum_{i \in I'} w_i g_i^+ \left(x, y^k\right) \tag{7.36a}$$

$$\text{subject to } h\left(x, y^k\right) = 0 \tag{7.36b}$$

$$g_i\left(x, y^k\right) \leq 0, \quad i \in I \tag{7.36c}$$

and it is solved at any one time.

To include all mentioned possibilities Fletcher and Leyffer (1994) formulated a general feasibility problem (FP) defined as

$$\min_{x \in X} \sum_{i \in I'} w_i g_i^+ \left(x, y^k\right) \tag{7.37a}$$

$$\text{subject to } h\left(x, y^k\right) = 0 \tag{7.37b}$$

$$g_i\left(x, y^k\right) \leq 0, \quad i \in I \tag{7.37c}$$

The weights w_i are non-negative and not all are zero. Note that with $w_i = 1 \;\; i \in I'$ we obtain the l_1-minimization. Also in the l_∞-minimization, nonnegative weights are existing at the solution such that

$$\sum_{i \in I'} w_i = 1 \tag{7.38}$$

and $w_i = 0$ if $g_i\left(x, y^k\right)$ does not reach the maximum value.

Note that infeasibility in the primal problem is detected when a solution of (FP) is obtained for which its objective value is greater than zero.

The solution of the feasibility problem (FP) gives information on the Lagrange multipliers for the equality and inequality constraints that are denote as $\bar{\lambda}^k, \bar{\mu}^k$ respectively. Then, the Lagrange function resulting from on infeasible primal problem at iteration k can be defined as

$$\bar{L}^k\left(x, y, \bar{\lambda}^k, \bar{\mu}^k\right) = \bar{\lambda}^{k^T} h(x, y) + \bar{\mu}^{k^T} g(x, y) \tag{7.39}$$

Remark 2

It should be noted that two different types of Lagrange functions are defined depending on whether the primal problem is feasible or infeasible. In addition, the upper bound is obtained only from the feasible primal problem.

The master problem

The derivation of the master problem in the GBD utilizes nonlinear duality theory and is characterized by the following three key ideas:

i. Projection of problem (7.23a–7.23e) onto the y-space;

ii. Dual representation of v; and

iii. Dual representation of the projection of problem (7.23a–7.23e) on the y-space.

In the sequel, the theoretical analysis involved in these three key ideas is presented.

i. Projection of (7.23a–7.23e) onto the ***y***-space

Problem (7.23a–7.23e) can be written as

$$\min_{y} \ \inf_{x} f(x,y) \tag{7.40a}$$

$$\text{subject to } \ h(x,y)=0 \tag{7.40b}$$

$$g(x,y)\le 0 \tag{7.40c}$$

$$x\in X \subseteq \Re^{n} \tag{7.40d}$$

$$y\in Y=\{0,1\}^{q} \tag{7.40e}$$

where the min operator has been written separately for y and x. Note that it is infimum with respect to x since for given y the inner problem may be unbounded. Let us define $v(y)$

$$v(y)=\inf_{x} f(x,y) \tag{7.41a}$$

$$h(x,y)=0 \tag{7.41b}$$

$$g(x,y)\le 0 \tag{7.41c}$$

$$x\in X$$

Note that $v(y)$ is parametric in the y variables and therefore, from its definition corresponds to the optimal value of problem (7.23a–7.23e) for fixed y (i.e., the primal problem $P(y^k)$ for $y = y^k$).

Figure 7.5 shows the flowchart for solving GBD method.

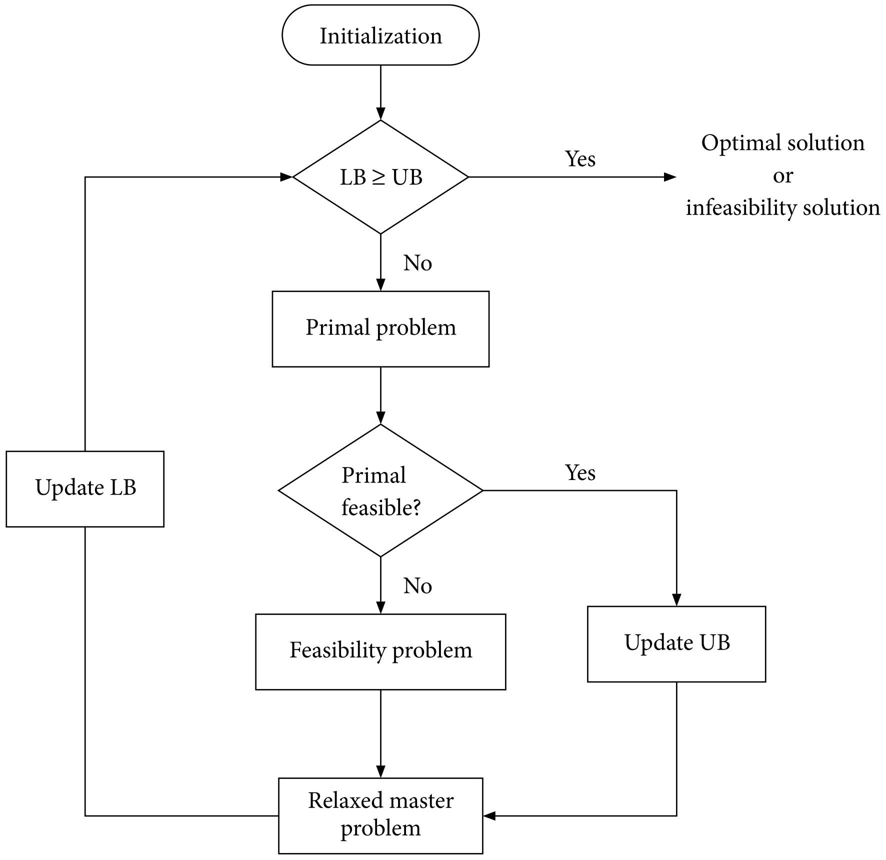

Fig. 7.5 Generalized Benders Decomposition method

7.2.2.3 Convergence and termination criteria

The convergence criterion of Generalized Benders Algorithm's as proposed by Geoffrion (1972) is:

The GBD algorithm terminates with an upper bound representing the optimal solution when the optimal value of the master problem approaches from below the upper bound given by the primal, within a convergence error ε.

The solutions of the master problem may go beyond the values provided by the primal problem during the implementation of the Generalized Benders Decomposition. This occurs as the Property (P) and/or L-Dual-Adequacy do not hold. Therefore, to tackle these situations a termination criterion is added. The new termination criterion is given as follows: [http://dx.doi.org/10.1016/0098-1354%2891%2985015-M]

The algorithm terminates with the upper bound representing the optimal solution, when the optimal value of the master problem, $\hat{y}^0$, go above the current upper bound (Floudas *et al.* 1989).

Whenever the criterion for termination is not activated and the relaxed master sequence of the NP-GBA converges, the converged value of the master problem is equal to the upper bound achieved from the solution of the primal. This is also true for non-convex problems which may show a gap between $v(y)$ and its dual. To be sure, after convergence of the relaxed master sequence, the Kuhn-Tucker conditions of the corresponding primal imply that $u^{(k_1)T} G\left(x^{(k_1)}, y\right) = 0$. In turn, this shows that one of the master's constraints becomes $y_0 \geq F(x^{(k_1)}, y)$. The criterion for termination is not activated and thus, $y_0 \geq F(x^{(k_1)}, y)$ will render $y_0 = F(x^{(k_1)}, y)$.

Application

Mixed-integer nonlinear programming (MINLP) has a wide range of applications for process design problems that allow simultaneous optimization of the process configuration as well as operating conditions (Floudas, 1995). A mixed-integer nonlinear programming (MINLP) is proposed for designing the optimal configuration of membrane networks to separate multi-component gas mixtures based on an approximate permeator model. The MINLP design model is developed for minimizing the total annual cost of the process by simultaneous optimization of the permeator configuration and operating conditions [Qi and Henson, (2000)].

The nonlinear nature of these mixed-integer optimization problems may arise from (i) nonlinear relations in the integer domain exclusively (e.g., products of binary variables in the quadratic assignment model), (ii) nonlinear relations in the continuous domain only (e.g., complex nonlinear input-output model in a distillation column or reactor unit), (iii) nonlinear relations in the joint integer-continuous domain (e.g., products of continuous and binary variables in the scheduling/planning of batch processes, and retrofit of heat recovery systems). Synthesis of reactor network and heat exchanger network, design of optimal distillation tower are common application of MINLP in chemical industry.

Summary

- Discrete variables like standard pipe diameter, number of HE in a HEN, number of plates in the distillation column, etc., are used to develop optimization problem. Optimization methods with discrete variables are discussed in this chapter. An optimization problem was divided into N stages and solved using dynamic programming algorithm. A problem of equipment replacement in chemical industry is represented as a discrete process; where each year is considered as a stage. Generalized Benders Decomposition method was used to solve an MINLP problem. The flowchart, algorithm, and termination criteria have been considered in this chapter.

Exercise

7.1 Find the value of x_1 and x_2, which minimizes

$$F = x_1^2 + x_2^2 - x_1 x_2 - 3x_2$$

subject to

$$-x_1 - x_2 \geq -2$$

$$x \geq 0\ ,\ x_2 \text{ integer}$$

7.2 The example proposed by Yuan *et al.* (1988). It involves three continuous variables and four binary variables. The formulation is

$$\min_{x,y} (y_1 - 1)^2 + (y_2 - 2)^2 + (y_3 - 1)^2 - \ln(y_4 - 1)$$

$$(x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2$$

subject to

$$y_1 + y_2 + y_3 + x_1 + x_2 + x_3 \le 5$$

$$y_1^2 + x_1^2 + x_2^2 + x_3^2 \le 5.5$$

$$y_1 + x_1 \le 1.2$$

$$y_2 + x_2 \le 1.8$$

$$y_3 + x_3 \le 2.5$$

$$y_4 + x_1 \le 1.2$$

7.3 A linear programming problem with *n* decision variables and *m* constraints can be considered as an *n*-stage dynamic programming problem with *m* state variables-Justify this statement with proper example.

7.4 Minimize the function (using dynamic programming)

$$F = (50x_1 - 0.2x_1^2) + (50x_2 - 0.2x_2^2) + 8(x_1 - 80)$$

subject to the constraints

$$x_1 \ge 75$$

$$x_1 + x_2 = 220$$

$$x_1, x_2 \ge 0$$

7.5 What is the difference between an initial value problem and a final value problem?

7.6 A refinery needs to supply 50 barrels of petrol at the end of the first month, 80 barrels at the end of second month, and 100 barrels at the end of third month. The production cost of x barrels of petrol in any month is given by \$($450x + 20x^2$). It can produce more petrol in any month and supply it in the next month. However, there is an inventory carrying cost of \$10 per barrel per month. Find the optimal level of production in each of the three periods and the total cost involved by solving it as an initial value problem.

7.7 Find the maximum value of the function

$$F = x_1 x_2$$

subject to the constraints

$$x_1^2 + x_2^2 \le 4$$

$$x_1, x_2 \ge 0 \text{ and integer}$$

7.8 Discuss the termination criteria of GBD method.

References

Beale, E. M. L. 'Discrete Optimization II' *Branch and Bound Methods for Mathematical Programming Systems Annals of Discrete Mathematics* 5(1979): 201–19.

Bellman, R. 1953. *An Introduction to the Theory of Dynamic Programming,* The RAND Corporation, Report R–245.

Bellman, R. 1954. *The Theory of Dynamic Programming,* Bull. Am. Math. Soc., 60: 503–16.

Benders, J. F., *Partitioning Procedures for Solving Mixed-variables Programming Problems,* Numerische Mathematik, 4(1962): 238–52.

Birewar, D. B. and Grossmann, I. E. 1990. *Simultaneous Production Planning and Scheduling in Multiproduct Batch Plants,* Ind. Eng. Chem. Res., 29: 570–80.

Duran, M. A. and Grossmann, I. E. *An Outer Approximation Algorithm for a Class of Mixed Integer Nonlinear Programs,* Mathematical Programming, 36(1986): 307–39.

Fletcher, R. and Leyffer S. 1994. *Solving Mixed Integer Nonlinear Programs by Outer Approximation.* Mathametical Programming, 66: 327–49.

Floudas, C. A. and Ciric, A. R. 1989. *Strategies for Overcoming Uncertainties in Heat Exchanger Network Synthesis,* Computers and Chemical Engineering, 13(10): 1133–52.

Floudas, C. A. and Grossmann, I. E. 1994. *Algorithmic Approaches to Process Synthesis: Logic and Global Optimization.* In Proceedings of Foundations of Computer-Aided Design, Colorado: FOCAPD '94, Snowmass.

Geoffrion, A. M., *Generalized Benders Decomposition, Journal of Optimization Theory and Applications* 10(4): 1972.

Grossmann, I. E. and Sargent, R. W. H. 1979. *Optimum Design of Multipurpose Chemical Plants,* Ind. Eng. Chem. Process Des. Dev., 18(2): 343.

Grossmann, I. E. 1990. *Mixed-integer Nonlinear Programming Techniques for the Synthesis of Engineering Systems.* Res. Eng. Des., 1: 205.

Grossmann, I. E. Quesada, I. Ramon, R. and Voudouris, V. T. 1992. *Mixed-integer Optimization Techniques for the Design and Scheduling of Batch Processes.* 'In Proc. of NATO Advanced Study Institute on Batch Process Systems Engineering '.

Holmberg, K. 1990. *On the Convergence of Cross Decomposition,* 'Mathematical Programming', 47: 269–96.

Kalitventzeff, B. and Maréchal, F. 1988. *The Management of a Utility Network,* Process System Engineering.

Kalitventzeff, B. 1991. *Mixed Integer Non-linear Programming and its Application to the Management of Utility Networks*, 'Engineering Optimization' vol. 18(1–3): 183–207.

Kocis, G. R. and Grossmann, I. E. 1869. *Relaxation Strategy for the Structural Optimization of Process Flow Sheets,* Ind. Eng. Chem. Res., 26, 1987.

Kocis, G. R. and Grossmann, I. E. *Global Optimization of Non-convex Mixed-integer Nonlinear Programming (MINLP) Problems in Process Synthesis.* 'Industrial Engineering Chemistry Research', 27(1988): 1407–21.

Kokossis, C. Floudas, C. A. 1990. *Optimization of Complex Reactor Networks—I.* ' Isothermal operation Antonis Chemical Engineering Science', 45(3): 595–614.

Qi, R. Henson, M. A. *Membrane System Design for Multi-component Gas Mixtures via Mixed-integer Nonlinear Programming,* 'Computers and Chemical Engineering', 24(2000): 2719–37.

Reklaitis, G. V. 1991. *Perspectives on Scheduling and Planning of Process Operations,* 'Presented at the Fourth international symposium on process systems engineering', Canada: Montebello.

Viswanathan, J. and Grossmann, I. E. *A Combined Penalty Function and Outer–Approximation Method for MINLP Optimization,* Computers and Chemical Engineering, 14(1990): 769–82.